Journal of Information Technology Education: Research

An Official Publication of the Informing Science Institute
InformingScience.org

JITEResearch.org

Volume 25, 2026

From Moodle Logs to Cognitive Load and Performance Insights: A Theory-Driven Analytics Methodology

Thaddeus Ogwoka*	Technical University of Mombasa, Mombasa, Kenya	togwoka@tum.ac.ke
Robert Oboko	University of Nairobi, Nairobi, Kenya	roboko@uonbi.ac.ke
Christopher Chepken	University of Nairobi, Nairobi, Kenya	chepken@uonbi.ac.ke

* Corresponding author

Abstract

Aim/Purpose	The study proposes and evaluates a theory-driven analytics methodology for inferring learners' cognitive load and performance from Moodle log data in online learning environments.
Background	Although Moodle captures rich learner interaction data, existing analytics largely focus on descriptive engagement metrics and provide limited insight into learners' cognitive processes. This gap restricts the ability of learning analytics to support cognitively informed instructional design and adaptive learning.
Methodology	An experimental design was employed, with 324 undergraduate students from three faculties in a public University, randomly assigned to treatment and control groups. The treatment group received cognitive load-informed interface scaffolds, while the control group accessed the same content without interventions. Moodle log data collected over four weeks were analyzed using statistical tests, clustering, classification, and association rule mining within a nine-step analytics workflow implemented in Python.
Contribution	The study introduces a reproducible analytics methodology that explicitly embeds extraneous, intrinsic, and germane cognitive load constructs into Moodle log analysis, extending existing LMS analytics frameworks from descriptive engagement monitoring to theory-grounded explanatory and predictive analysis.

Accepting Editor Bahman Gorjian │ Received: October 14, 2025 │ Revised: January 25, 2026 │ Accepted: February 10, 2026.

Cite as: author. (2026). From Moodle logs to cognitive load and performance insights: A theory-driven analytics methodology. *Journal of Information Technology Education: Research, 25,* Article 19. https://doi.org/10.28945/5723

Findings	Results indicate that treatment learners demonstrated significantly higher engagement and performance than control learners, with large effects for course activity completion (d = 0.77) and module views (d = 0.65), and moderate effects for quiz grades (d = 0.38). Machine learning analyses further revealed more differentiated behavioral patterns and stable associations between cognitive load-related behaviors and learning outcomes among treatment learners.
Recommendations for Practitioners	Educators and instructional designers can apply the proposed methodology to unobtrusively monitor learners' cognitive load and implement targeted interface scaffolds that enhance engagement and performance without altering core instructional content.
Recommendations for Researchers	Researchers are encouraged to adopt and extend the proposed methodology to examine cognitive load dynamics across disciplines, platforms, and instructional designs, and use it as a foundation for theory-driven learning analytics studies that move beyond purely predictive models.
Impact on Society	By enabling cognitively informed learning analytics at scale, this methodology supports more effective and equitable online education, particularly in resource-constrained contexts.
Future Research	Future work should validate log-based cognitive load indicators using multimodal data, explore real-time adaptive learning systems responsive to inferred cognitive load, and assess the long-term effects of cognitive load-informed interventions on learner retention and achievement.

Keywords	cognitive load, learner performance, learning analytics, Moodle log data, online learning, learning management system, instructional design

INTRODUCTION

The rapid expansion of online learning has positioned Learning Management Systems (LMS) such as Moodle at the center of instructional delivery in higher education (Rachel et al., 2018). Moodle is one of the most widely adopted LMS platforms globally, offering tools for content delivery, assessment, and learner tracking (Krishnan et al., 2022). As online learning scales, understanding how learners interact with LMS environments and how these interactions relate to learning effectiveness has become a central concern for educators and researchers alike (Tlili et al., 2023).

To address this need, learning analytics has increasingly relied on Moodle log data, which capture detailed traces of learner behavior, including course views, quiz attempts, activity completions, and time on task (Rotelli & Monreale, 2021). These data have been widely used to model engagement, predict academic performance, and identify at-risk learners (Conijn et al., 2017; Rachel et al., 2018). However, most Moodle analytics remain largely descriptive or predictive, focusing on *what* learners do rather than *why* they behave in certain ways or *how* these behaviors reflect underlying learning processes (Moodle, n.d.a).

A critical limitation of existing LMS analytics is their limited engagement with cognitive theory. In particular, cognitive load theory (CLT) provides a well-established framework for explaining how instructional design and task complexity influence learning by imposing extraneous, intrinsic, or germane cognitive load on learners (Sun et al., 2023; Sweller et al., 2011a). Excessive extraneous or intrinsic load can overwhelm working memory and impair learning, whereas germane load supports schema construction and deeper understanding. Despite its relevance, CLT has rarely been systematically integrated into Moodle-based analytics, leaving a gap between observed learner behavior and underlying cognitive mechanisms.

As a result, high levels of observable engagement, such as frequent clicks or extended time-on-task, are often interpreted as positive indicators, even though they may reflect disorientation or cognitive overload rather than effective learning. While some studies have examined cognitive load in online learning contexts, they typically rely on self-report instruments, expert judgments, or small-scale experimental settings, limiting their applicability to large-scale LMS environments (Bănuț & Andronache, 2023; Yen et al., 2015). Few studies have proposed a systematic methodology that uses Moodle log data as theoretically grounded proxies for extraneous, intrinsic, and germane cognitive load and directly links these measures to learner performance.

This study addresses this gap by proposing and empirically testing a structured methodology that integrates cognitive load theory into Moodle learning analytics. Specifically, it introduces a nine-step analysis framework that extends prior LMS analytics approaches by systematically mapping Moodle log indicators such as course views, quiz interactions, activity completion, and time-on-task to extraneous, intrinsic, and germane cognitive load dimensions, alongside learner performance outcomes. By embedding cognitive load constructs directly into Moodle log analysis, the proposed methodology provides a theoretically grounded and scalable approach for interpreting learner behavior and supporting data-driven instructional design. To situate this contribution, the following literature review examines prior work in learning analytics and cognitive load theory, with particular attention to how learner behavior has been analyzed in Moodle learning environments, the limitations of existing cognitive load measurement approaches, and the conceptual foundations required to link LMS log data to cognitive load dimensions systematically.

Literature Review

Learning Analytics in Online Learning Environments

Learning analytics has made substantial progress in leveraging LMS log data to describe and predict learner behavior and performance in online learning environments. Foundational frameworks developed within the Learning Analytics and Knowledge (LAK) and Society for Learning Analytics Research (SOLAR) communities conceptualize learning analytics as a cyclical process involving data collection, analysis, interpretation, and action (Shum & Ferguson, 2012; Siemens & Long, 2014). Within this cycle, learner interaction data captured by Learning Management Systems (LMSs) such as Moodle have been widely used to study engagement, persistence, and academic performance.

Moodle, in particular, generates fine-grained system logs that record learners’ interactions with instructional materials, assessments, and communication tools. Prior studies have leveraged these logs to predict outcomes such as grades, course completion, and dropout risk (Fenu et al., 2017; Krishnan et al., 2022; Rachel et al., 2018). While these approaches have demonstrated the predictive value of LMS data, they largely focus on behavioral frequency and outcome prediction, often treating the learning process as a “black box” without explicit reference to underlying cognitive mechanisms.

As a result, although LMS analytics effectively capture what learners do, they provide limited insight into how learners cognitively process instructional materials or why particular interaction patterns lead to success or failure. This limitation has motivated growing interest in the theoretically grounded learning analytics approaches that connect observable learner behavior with established cognitive theories. Among these, cognitive load theory offers a particularly relevant framework for interpreting learners’ cognitive processing and learning outcomes.

Cognitive Load Theory in Online Learning

Cognitive load theory (CLT) explains how learning occurs under constraints of human working memory and offers a powerful lens for interpreting learner behavior in online learning environments (Orru & Longo, 2019). Building on foundational work by Miller and later formalized by Sweller, CLT distinguishes three types of cognitive load that shape learning outcomes: intrinsic load, which arises from the inherent complexity of the learning material; extraneous load, which results from suboptimal instructional design or interface features; and germane load, which reflects the cognitive resources devoted for schema construction and knowledge integration(Skulmowski & Xu, 2022; Sun et al., 2023). In online learning systems such as Moodle, these load types are not directly observable but are instead inferred from learners' interaction patterns with instructional content and activities.

In online learning environments, poorly designed interfaces, excessive navigation, and redundant media can increase extraneous cognitive load, thereby impairing performance (Abeysekera et al., 2024; Clark & Mayer, 2016). Conversely, instructional scaffolds that structure content, support collaboration, or encourage self-explanation can help manage intrinsic load and foster germane load. Importantly, these design effects are manifested in learners' observable behaviors such as navigation frequency, repeated content access, quiz engagement, and task completion, making them potentially detectable through LMS log data. Although numerous experimental studies have validated these principles in controlled settings, many rely on self-report questionnaires or small-scale laboratory designs (Sweller, 2018).

Although self-reported measures remain the most common method for measuring cognitive load, they are intrusive, retrospective, and poorly suited for continuous use in authentic online learning environments. As a result, the practical integration of cognitive load theory into large-scale learning analytics systems has remained limited, despite its strong explanatory value. This methodological constraint has prompted increasing interest in unobtrusive, behavior-based approaches that infer cognitive load from learners' interactions with learning management systems, creating a need for systematic frameworks that connect cognitive load theory with LMS log data.

Measuring Cognitive Load Beyond Self-Report

Beyond self-report instruments, cognitive load has been measured using physiological indicators such as eye tracking and electrocardiography, dual-task methods, and performance-based metrics (Minkley et al., 2021). While these approaches offer greater measurement precision, they are resource-intensive and impractical for deployment in real-world LMS contexts involving large student cohorts.

Consequently, research has increasingly explored the use of LMS behavioral data as indirect, system-level proxies for cognitive processing. Yen et al. (2015) demonstrated that learner interaction patterns in online courses can reflect cognitive conditions and inform adaptive instructional strategies. Similarly, Cunha and Figueira (2021) analyzed sequences and durations of LMS activities to infer learner engagement and cognitive effort. Krishnan et al. (2022) further showed that log-based indicators such as content access and assessment behavior can distinguish active from disengaged learners and support instructional improvement.

Collectively, these studies suggest that, while indirect, LMS logs provide ecologically valid and scalable indicators of learners' interaction with instructional materials, making them suitable for cognitive load-informed analytics when grounded in appropriate theory.

Conceptual Foundations for Linking LMS Activities to Cognitive Load Dimensions

A key conceptual bridge between LMS analytics and cognitive load theory is provided by models that interpret the depth of learner interaction. The Moodle cognitive depth indicator model, derived from the community of inquiry framework (Garrison et al., 1999; Moodle, n.d.b), conceptualizes learner activity along increasing levels of engagement, ranging from passive content exposure to active contribution and feedback integration. Higher levels of cognitive depth represent more intentional, constructive, and self-regulated learning behaviors.

Although originally developed to capture cognitive presence rather than cognitive load directly, this hierarchical activity model aligns closely with CLT principles. Shallow interaction patterns may signal cognitive disorientation or excessive extraneous load, whereas deeper engagement and task completion behaviors are more consistent with germane cognitive processing and schema construction. Prior Moodle analytics studies have implicitly relied on similar assumptions when interpreting engagement patterns, even when cognitive load was not explicitly modeled (Álvarez-Méndez et al., 2020; Blayney et al., 2015).

Limitations of Existing Moodle Analytics Methodologies

Despite advances in Moodle-based learning analytics, several limitations persist. First, many frameworks prioritize predictive accuracy over interpretability, offering limited theoretical explanation of why certain behaviors predict performance (Rachel et al., 2018). Second, cognitive load is rarely treated as a core construct, even though it is known to strongly influence learning outcomes. Third, existing methodologies often focus on isolated analytics techniques such as prediction or clustering without integrating them into a coherent, theory-driven workflow.

Moreover, few studies critically address the challenges of inferring cognitive states from behavioral data, including ambiguity in interpretation and potential confounds (e.g., persistence versus efficiency). As a result, a gap remains between cognitive theory, LMS analytics practice, and scalable methodological integration.

Research Gap

Taken together, the literature reveals a clear disconnect between cognitive load theory and LMS analytics research. While cognitive load has been shown to play a central role in learning effectiveness, and Moodle logs capture rich traces of learner behavior, existing approaches have not systematically integrated cognitive load constructs into Moodle analytics frameworks. Most studies either rely on subjective measures of cognitive load or analyze LMS data without explicit cognitive grounding.

This gap underscores the need for a structured, scalable methodology that embeds cognitive load theory directly into Moodle log analysis, enabling researchers and practitioners to interpret learner behavior in terms of underlying cognitive processes rather than surface-level engagement alone.

Study Contribution

Building on the identified gap, this study proposes a structured nine-step methodology that integrates cognitive load theory into Moodle learning analytics. Unlike existing frameworks that emphasize engagement and performance alone (e.g., Fenu et al., 2017; Rachel et al., 2018), the proposed approach explicitly maps Moodle log indicators to extraneous, intrinsic, and germane cognitive load alongside learner performance metrics. This integration enables analysis not only of what learners achieve but also of how their cognitive processing influences learning outcomes.

By embedding cognitive load constructs directly into Moodle log analysis, the proposed methodology offers a theoretically grounded and scalable framework for analyzing learner behavior in authentic online learning environments, thereby advancing both learning analytics research and instructional design practice.

Methods

Research Design and Participants

This study employed a controlled experimental design integrated with a theory-driven learning analytics methodology to examine how Moodle log data can be systematically operationalized to reflect cognitive load dimensions and learner performance. Extending the prior Moodle analytics framework proposed by Rachel et al. (2018), the study explicitly embeds cognitive load theory into LMS-based analytics through a structured nine-step workflow that links learner behavior, cognitive processing, and performance outcomes.

By combining experimental manipulation with log-based analytics, the design enables empirical examination of whether theoretically targeted cognitive load interventions produce observable changes in learner interaction patterns and performance indicators.

The study was conducted at a public university in Africa during a four-week Moodle-based course. The target population comprised 842 undergraduate students enrolled in six computing-related programs. These programs were selected because students had prior exposure to Moodle, ensuring baseline familiarity with online learning environments and minimizing confounding effects related to the platform novelty.

Sample size was determined using Cochran's formula with a 95% confidence level, 5% margin of error, and a conservative 50% response distribution (Casteel & Bridier, 2021), yielding a required minimum of 385 participants. To ensure proportional representation across programs, stratified random sampling was employed (Cohen et al., 2007). A total of 324 students consented and participated in the study. Although slightly below the calculated threshold, this sample size exceeds minimum recommendations for controlled experimental studies and remains adequate for detecting group-level effects and behavioral patterns (Cohen et al., 2007).

Random Assignment and Experimental Conditions

Participants were randomly assigned to either a treatment or a control group using Microsoft Excel's RAND function, ensuring equal probability of allocation and minimizing selection bias (Ahmed, 2024). Randomization was conducted after consent and before course commencement, thereby preserving internal validity.

Both groups undertook the same four-week Moodle Lara Vel course, covering identical content, assessments, timelines, and grading criteria. The sole difference lay in the instructional scaffolds provided at the interface level for the treatment group. This isolation of the interface as the only manipulated factor strengthens causal inference by ensuring that observed differences in learner behavior, cognitive load indicators, and performance outcomes are attributable to the cognitive load management interventions rather than instructional or content variation.

Cognitive load management interventions

The treatment group accessed a customized Moodle interface that incorporated the BookActivity and Bootstrap Elements plugins. These modifications were grounded in cognitive load theory and designed to target the three cognitive load dimensions through interface-level scaffolding:

- *Extraneous cognitive load reduction:* Prompt links to worked-out exercises minimized unnecessary search and navigation demands, consistent with the worked-out example (Sweller et al., 2019).
- *Intrinsic cognitive load management:* Structured forum prompts supported collaborative problem solving and leveraged the collective working memory effect (Orru & Longo, 2019).
- *Germane cognitive load enhancement:* Links to partially worked examples encouraged self-explanation and schema construction, drawing on the expert reversal effect (Clark & Mayer, 2016).

The interventions were reviewed and validated by a focus group comprising instructors and instructional designers to ensure theoretical alignment and instructional feasibility. These interventions are central to the study because they function as deliberate, theory-driven manipulations intended to influence learners' cognitive load states, thereby enabling validation of whether Moodle log indicators sensitively reflect changes in extraneous, intrinsic, and germane cognitive load. Table 1 summarizes the conceptual mapping between interface modifications and targeted cognitive load dimensions.

Table 1. Mapping of interface modifications to cognitive load dimensions

Interface modification	Targeted cognitive load type	Conceptual rationale	Expected effect
Prompt links to worked-out examples (van Mierlo et al., 2012).	Extraneous load	Worked-example effect (Sweller et al., 2019) minimizes unnecessary cognitive processing.	High engagement, low extraneous load.
Embedded forum prompts for collaborative discussion (Álvarez-Méndez et al., 2020).	Intrinsic load	The collective working memory effect (Orru & Longo, 2019) enables distributed processing.	Tasks seem easy, low intrinsic load.
Prompt links to exercises with partial solutions (Renkl & Atkinson, 2002).	Germane load	The expert reversal effect (Clark & Mayer, 2016) supports schema construction.	Schema created, highly germane load.

Intervention provision procedure

As illustrated in Figure 1, interventions were introduced sequentially across three weeks to isolate their effects. In week 1, extraneous cognitive load (ECL) scaffolds were deployed; in week 2, intrinsic cognitive load (ICL) scaffolds; and in week 3, germane cognitive load (GCL) scaffolds. Moodle log data were collected at the end of each week to capture learner interaction patterns corresponding to the specific cognitive load dimension targeted during that phase.

The control group interacted with the standard LMS interface throughout the experimental period. In week 4, all interventions were made available to the control group to ensure ethical equity and prevent long-term instructional disadvantage.

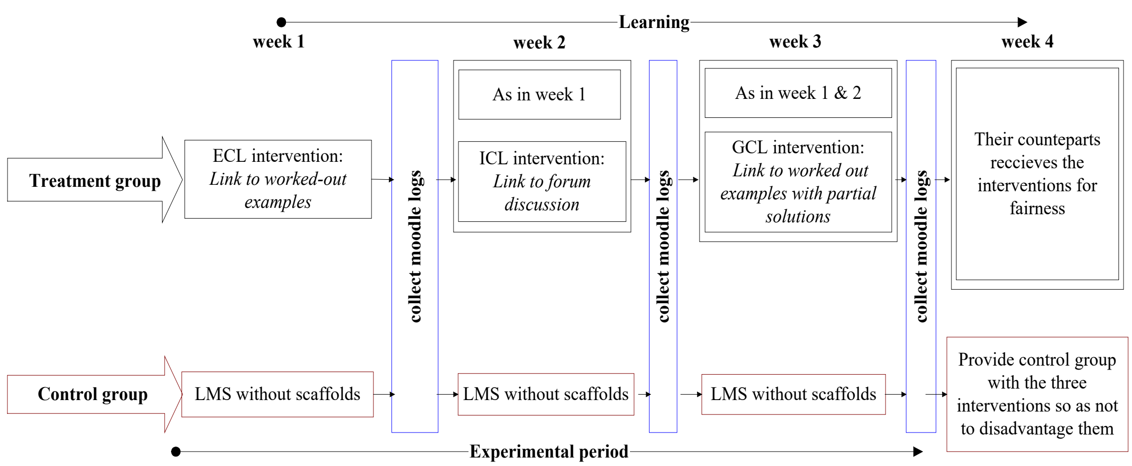

Figure 1. Cognitive load interventions: provisions procedure

Measurement and use of Moodle log proxies

Cognitive load was inferred using behavioral indicators extracted from Moodle system logs, including course views, quiz attempts, quiz submissions, time-on-task, activity completions, and grades. These indicators were selected based on prior learning analytics research demonstrating their relevance as scalable, ecologically valid proxies for learner cognitive processing in authentic LMS contexts.

Unlike self-report instruments, which capture subjective perceptions retrospectively, log-based indicators reflect actual learner behavior as it occurs during learning activities. Table 2 presents the operational mapping between Moodle indicators, cognitive load dimensions, and performance outcomes.

Table 2. Cognitive load and performance measurement scale

Moodle indicator	Cognitive load type and performance	High count 🟢	Low count 🔴
Course views	Extraneous load	🟢 High engagement → Low extraneous load	🔴 Low engagement → High extraneous load
Quiz attempt views	Intrinsic load	🟢 Tasks seem easy → Low intrinsic load	🔴 Task difficulty → High intrinsic load
Quiz attempt submissions	Germane load	🟢 Schema created → High germane load	🔴 Limited schema → Low germane load
Course activity completions	Performance	🟢 Better performance	🔴 Low performance
Grade/30	Performance	🟢 Better performance	🔴 Low performance

Performance is treated as a complementary analytic construct rather than a cognitive load dimension, enabling integrated examination of how variations in extraneous, intrinsic, and germane load manifest in observable learning outcomes within Moodle.

It is acknowledged that these indicators represent indirect measures of cognitive load. However, when grounded in theory and interpreted collectively rather than in isolation, they provide meaningful system-level insights consistent with established LMS analytics practices.

To distinguish "high" and "low" levels of engagement, relative thresholds were used rather than fixed cut-off values. Specifically, log counts above the group median were classified as high engagement, while those below the median were classified as low engagement. This distribution-based approach avoids arbitrary thresholds and aligns with prior analytics research.

Limitations of Moodle log proxies

Moodle log indicators are proxy measures rather than direct measurements of cognitive load. For example, high quiz submission frequency may reflect productive germane processing but may also indicate persistence rather than efficiency. While physiological and dual-task measures offer high precision (Minkley et al., 2021), they are impractical for large-scale LMS contexts. Consistent with prior studies such as Yen et al. (2015), this research adopts Moodle logs as pragmatic, system-level indicators that enable scalable and unobtrusive analysis of cognitive processing patterns in authentic online learning environments. Convergent evidence from theoretically targeted interventions and corresponding log-based behavioral changes provides indirect empirical support for the validity of these proxies.

Ethical considerations and participant protection

Ethical approval for the study was obtained from the National Commission for Science, Technology, and Innovation (License No. NACOSTI/P/23/30310. All participants were informed about the study objectives, procedures, and their right to withdraw at any time without specific penalty. Consent was obtained electronically before participation, and all identifying information was anonymized before analysis.

To ensure fairness, both treatment and control groups received identical content, assessments, and grading criteria. The only difference was the presence of instructional scaffolds in the treatment interface. After completion of the experimental phase, the enhanced interface was made available to all learners.

DATA COLLECTION OVERVIEW

At the end of the course, learner interaction data were extracted from Moodle system logs in Excel format. The logs captured time-stamped records of learner interactions, including course views, quiz activities, time spent on tasks, activity completion, and grades. These data constituted the raw input for the proposed nine-step analytics methodology, which transforms system logs into theoretically grounded cognitive load and performance indicators.

Data analysis was conducted in Python, with detailed preprocessing procedures, analytic workflows, and code excerpts provided in the appendices to support reproducibility without overloading the main text.

PROPOSED ANALYTICS METHODOLOGY

While the preceding sections describe the experimental context and measurement strategy, the central contribution of this study is a nine-step analytics methodology that systematically integrates cognitive load theory into Moodle learning analytics. The methodology provides a structured, reproducible workflow for transforming raw Moodle log data into theoretically grounded indicators of extraneous, intrinsic, and germane cognitive load, alongside learner performance outcomes.

Rather than treating Moodle logs as purely descriptive engagement traces, the proposed approach embeds cognitive theory directly into the analytics pipeline, extending existing frameworks such as Rachel et al. (2018). For clarity and parsimony, the nine steps are organized into three analytic phases (Figure 2): data preparation, theory-driven construct operationalization, and multi-level analysis.

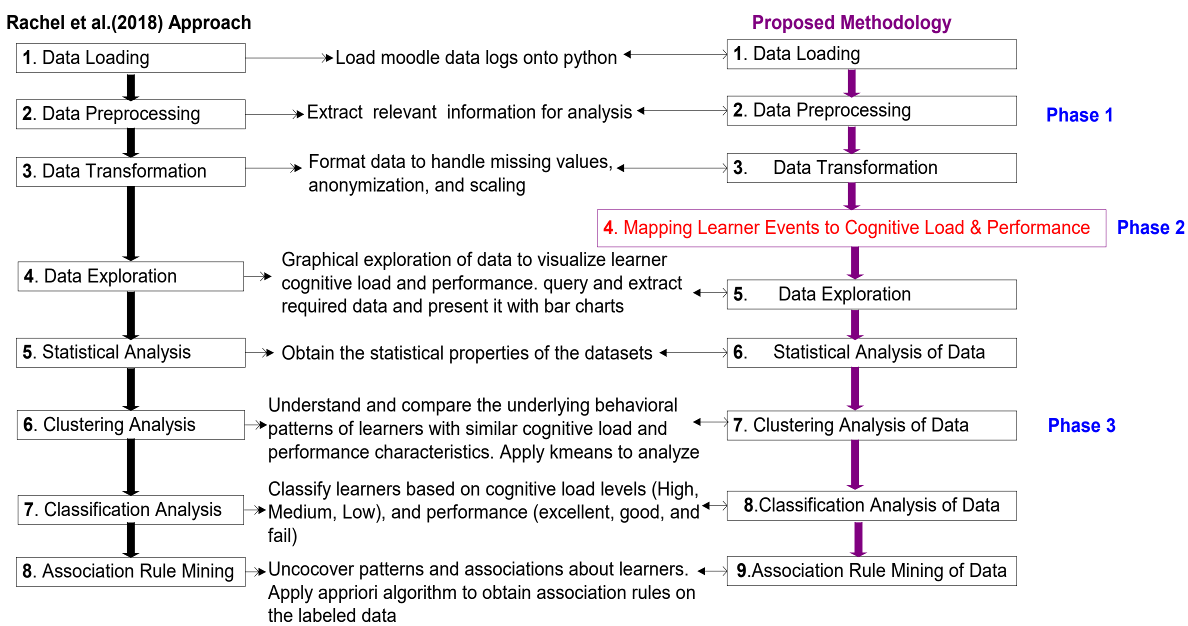

Figure 2. The proposed nine-step Moodle analytics methodology

Phase 1: Data preparation and structuring (Steps 1-3)

The first phase focuses on converting raw Moodle system logs into analyzable learner-level datasets.

Step 1: Data loading. Moodle logs capturing time-stamped learner interactions were extracted from the LMS and imported into the Python analytics environment.

Step 2: Data preprocessing. Records unrelated to learner activity (e.g., administrative and system events) were removed, duplicates eliminated, and time stamps standardized. Learner events were aggregated into structured indicators, including course views, quiz stamps, quiz submissions, time-on-task, activity completion, and grades.

Step 3: Data transformation. Missing values were addressed using mean substitution, with sensitivity checks using median replacement, confirming that alternative imputation strategies produced comparable results. Variables were transformed into consistent learner-level representations suitable for downstream analysis.

Detailed data loading, preprocessing, and transformation procedures, data schemas, and code implementations corresponding to steps 1-3 are provided in Appendices A-C to support reproducibility while maintaining methodological clarity in the main text.

Phase 2: Theory-driven mapping of Moodle logs to cognitive load constructs (Step 4)

The second phase constitutes the theoretical core of the methodology.

Step 4: Mapping learner events to cognitive load and performance constructs. Learner interaction indicators were mapped to cognitive load constructs based on cognitive load theory and prior learning analytics research. As summarized in Appendix D, course and module views were operationalized as indicators of extraneous cognitive load, quiz attempt behaviors as indicators of intrinsic cognitive load, and quiz submissions and time-on-task as indicators of germane cognitive load. Learner performance was measured using course activity completion and quiz grades.

This mapping is theoretically justified in three ways. First, it aligns with CLT's distinction between surface navigation, task engagement, and schema construction. Second, it is consistent with depth-of-engagement models used in Moodle analytics, where shallow interaction patterns reflect passive or disoriented engagement, and deeper task completion reflects constructive processing. Third, similar indicators have been validated in prior LMS-based studies as meaningful proxies for cognitive effort and learning processes (e.g., Blayney et al., 2015; Krishnan et al., 2022)

Although Moodle logs do not directly measure cognitive load, their use as proxy indicators offers a scalable, ecologically valid approach to studying cognitive processes in authentic online learning environments. This limitation is acknowledged; however, grounding the mappings in theory and applying them consistently across analyses strengthens their construct validity. Operational thresholds used to distinguish relatively "high" versus "low" interaction frequencies were derived empirically from the distributional properties of the log data and are reported in Appendix D to ensure transparency and reproducibility.

Learner performance is treated as an analytically distinct outcome construct rather than a cognitive load category, enabling examination of how theoretically grounded cognitive load indicators relate to observable learning outcomes.

Phase 3: Multi-level analysis and pattern discovery (Steps 5-9)

The third phase applies complementary analytical techniques to examine relationships between cognitive load and learner performance, with particular attention to differences between intervention (treatment) and non-intervention (control) conditions.

Step 5: Exploratory data analysis. Descriptive statistics and visualizations were used to summarize engagement patterns, cognitive load distributions, and initial differences between treatment and control groups.

Step 6: Statistical analysis. Inferential statistics tests were conducted to examine associations between cognitive load indicators and learner performance outcomes and to assess group-level differences.

Beyond traditional statistics, machine learning techniques were incorporated to uncover latent patterns and predictive relationships.

Step 7: Clustering analysis. Unsupervised clustering methods were applied to identify distinct learner profiles based on cognitive load and performance indicators.

Step 8: Classification analysis. Supervised classification models were developed to evaluate the extent to which cognitive load indicators predicted learner performance outcomes.

Step 9: Association rule mining. Association rule mining was used to identify frequent behavioral patterns characterizing treatment and control groups and their associated cognitive conditions.

Together, Steps 5-9 enable triangulation across descriptive, inferential, and predictive perspectives, strengthening the robustness and interpretability of findings. Detailed algorithm configurations, parameter settings, and evaluation procedures for these steps are documented in Appendices E-I.

RESULTS

Applying the proposed nine-step analytics methodology, results are presented to address the research questions concerning (i) differences in engagement and performance between treatment and control groups, and (ii) behavioral patterns associated with cognitive load-informed interface interventions. To improve clarity and conciseness, only primary findings are reported in the main text, with supporting analyses and secondary visuals in the appendices.

ENGAGEMENT AND PERFORMANCE DIFFERENCES

Learner engagement was examined using Moodle log indicators, including module views, course/module views, quiz attempts, quiz submissions, activity completion, and grades. As summarized in Table 5, the treatment group consistently demonstrated higher engagement and performance than the control group.

Table 5. Summary of engagement indicators (Cohen's d with 95% CIs)

Indicator	Cohen's d	95% CI lower	95% CI upper
Course module viewed	0.65	0.45	0.84
Course viewed	0.41	0.21	0.60
Quiz attempt started	0.29	0.09	0.48
Quiz attempt viewed	0.22	0.03	0.42
Quiz attempt submitted	0.18	-0.02	0.37
Course activity completion	0.77	0.57	0.97
Grade/30	0.38	0.18	0.57

A Mann-Whitney U test indicated a statistically significant overall difference between groups ($U = 24{,}145.0$, $p = 0.009$). Effect sizes ranged from small to large, with the strongest differences observed for course module views ($d = 0.65$, 95% CI [0.45,0.84]) and course completion ($d = 0.77$,95% CI [0.57,0.97]). Quiz grades also showed a moderate effect ($d = 0.38$, 95% CI [0.18,0.57]). Figure 3 presents the mean group differences for the core indicators. Detailed descriptive statistics, correlation matrices, and distribution plots are provided in Appendix E.

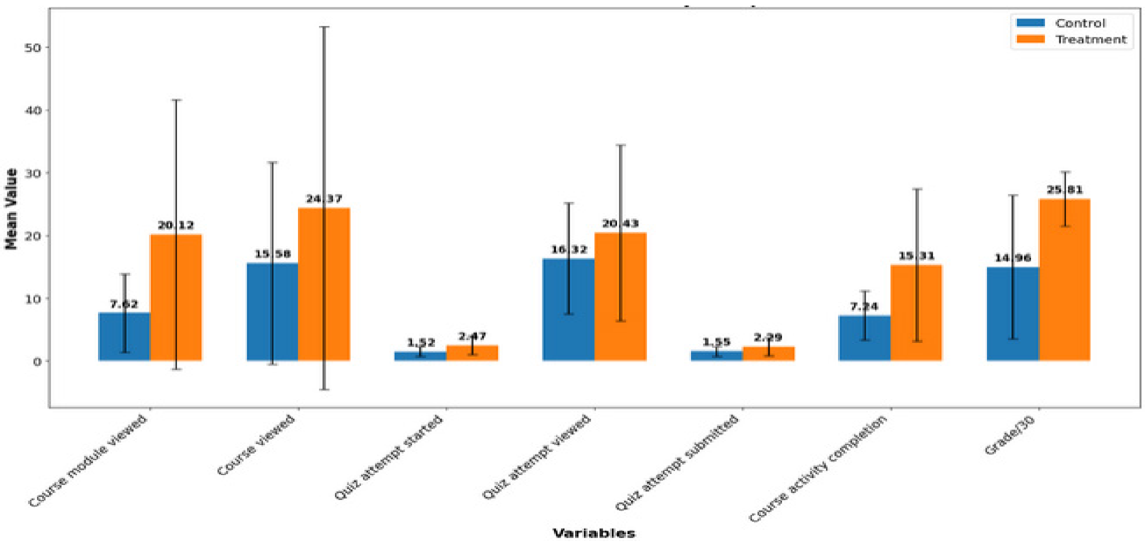

Figure 3. Mean differences in engagement between control and treatment groups

Cognitive Load-Related Indicators

Using the operational definitions in Table 2, engagement indicators were examined as proxies for cognitive load dimensions. Indicators associated with extraneous load (course and module views) and intrinsic load (quiz attempts started and viewed) were significantly higher in the treatment group. Indicators associated with germane (quiz submissions) showed smaller but consistent differences.

These indicators differed systematically between groups across all three cognitive load categories. No single indicator was interpreted in isolation; instead, patterns across indicators were considered collectively, consistent with the proposed methodology.

Behavioral Patterns: Clustering Analysis

Unsupervised k-means clustering was applied to identify learner behavioral profiles based on engagement, cognitive load indicators, and performance measures. The treatment group exhibited more clearly separated clusters than the control group, as reflected by higher silhouette scores (see Appendix G).

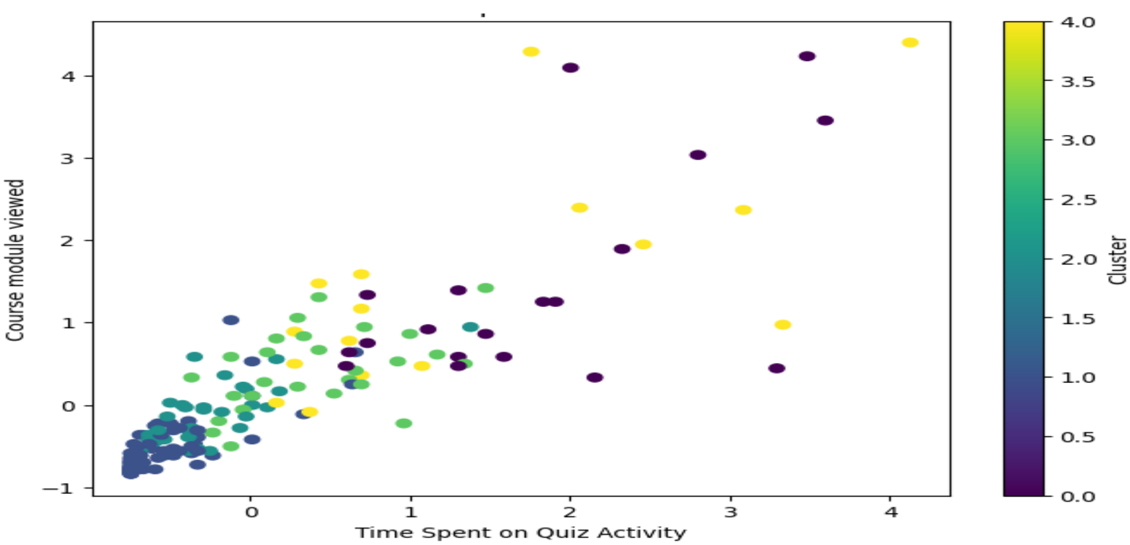

Figure 4. Treatment group clusters

Representative cluster visualizations for treatment and control groups are shown in Figures 4 and 5, respectively. The clustering results indicate greater behavioral differentiation among treatment learners compared to more homogeneous patterns in the control group.

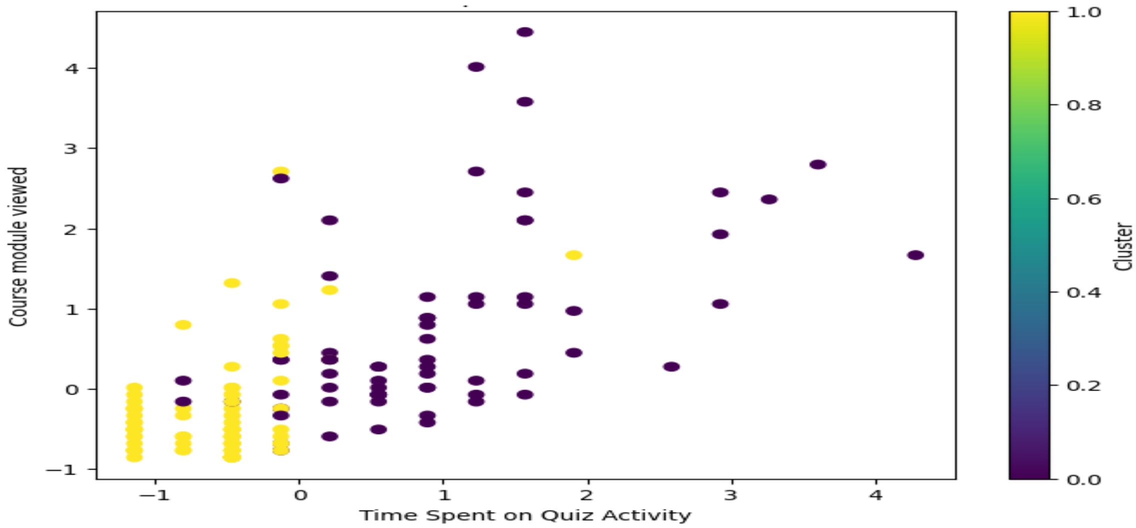

Figure 5. Control group clusters

Classification Analysis

Supervised classification models were developed to predict learner categories (absent, low, medium, high). Decision tree (DT), Random forest (RF), and Logistic regression (LR) models were evaluated using stratified 10-fold cross-validation.

As shown in Table 6, DT and RF models achieved high accuracy for the treatment group. Instances of 100% accuracy were observed in specific folds for tree-based models; however, these results were interpreted cautiously. Cross-validated ROC/AUC scores and out-of-sample validation confirmed that the model performance remained stable without evidence of systematic overfitting (full metrics reported in Appendix H).

Logistic regression produced lower but more conservative accuracy values, serving as a robustness benchmark. Together, these results indicate that cognitive load-related indicators have predictive utility, while acknowledging the limitations of perfect classification outcomes.

Table 6. Classification models' performance measures

	Treatment group				Control group			
Classifier	**Accuracy**	**Precision**	**Recall**	**F-1score**	**Accuracy**	**Precision**	**Recall**	**F-1 score**
DT	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0
RF	1.0	1.0	1.0	1.0	0.93	0.67	1.0	0.89
LR	0.87	0.96	0.93	0.9333	0.95	0.97	1.06	1.06

Association Rule Mining

Association rule mining was conducted to identify frequent behavioral patterns linked to activity completion. To reduce redundancy, only the strictest and most interpretable rules are presented in Table 7, with the full rule set reported in Appendix I.

The results show that a lack of course or module views strongly correlated with activity non-completion, while quiz submission was consistently associated with activity completion (confidence = 1.0, lift > 1.4). These rules highlight stable co-occurrence patterns between engagement behaviors and learning outcomes in the treatment group.

Table 7. Association rules between learner engagement behaviors and activity completion (treatment group)

Rule	Antecedent	Consequent	Support	Confidence	Lift
1	Course viewed=0	Activity not completed	0.60	1.00	1.67
2	Course module viewed=0	Activity not completed	0.60	1.00	1.67
3	Quiz attempt viewed = 1	Activity completed	0.60	1.00	1.43
4	Quiz attempt submitted=1	Activity completed	0.60	1.00	1.43
5	Quiz attempt started=0	Activity not completed	0.60	1.00	1.43

SUMMARY OF KEY FINDINGS

In summary, treatment learners demonstrated (i) higher engagement and performance, (ii) more differentiated behavioral patterns, and (iii) stronger predictive relationships between Moodle log indicators and learning outcomes than control learners. These findings directly address the research questions and provide empirical support for the proposed analytics methodology. Interpretive and theoretical implications are developed further in the discussion section.

DISCUSSION

This study demonstrates how cognitive load theory (CLT) can be operationalized within online learning environments through a theory-driven analytics methodology that maps Moodle log data to extraneous, intrinsic, and germane cognitive load constructs. Rather than treating learner interactions as purely descriptive engagement traces, the findings show that structured patterns of LMS behavior can be meaningfully interpreted as reflections of underlying cognitive processes when grounded in CLT.

INTERPRETING COGNITIVE LOAD THROUGH BEHAVIORAL PATTERNS

Differences between treatment and control groups indicate that interface-level scaffolding influenced not only the amount of learner interaction but also the organization of engagement behaviors. Higher course and module views, increased quiz attempts, and greater submission rates collectively suggest a shift from fragmented or disoriented navigation toward more purposeful task engagement. Importantly, these indicators were not interpreted as configurations of behavior, but rather as isolated metrics, acknowledging the indirect nature of log-based cognitive load proxies.

At the same time, alternative interpretations remain plausible. High interaction frequency may reflect persistence, repetition, or exploration rather than reduced extraneous load or optimized intrinsic load. By triangulating across navigation, task engagement, and completion indicators, the methodology reduces but does not eliminate this ambiguity. This reinforces the value of multi-indicator inference over single-metric conclusions when studying cognitive processes through LMS logs.

INSIGHTS FROM MACHINE LEARNING ANALYSES

The machine-learning analyses extend the statistical findings by revealing latent behavioral structures associated with the interventions. The emergence of more differentiated clusters within the treatment group suggests that cognitive-load-informed interfaces may enable learners to adopt diverse yet productive engagement strategies, rather than converging on uniformly low-effort or disengaged behaviors. In contrast, the relatively homogeneous clusters observed in the control group may reflect constrained interaction pathways imposed by the standard interface.

The high predictive accuracy of tree-based classifiers indicates strong associations between cognitive load-related indicators and learner performance outcomes. However, near-perfect classification accuracy must be interpreted cautiously. Rather than implying universal generalizability, these results likely reflect well-separated behavioral patterns induced by structured interface scaffolds. By incorporating cross-validation and more conservative baseline models, the study positions machine learning not as a replacement for theory, but as a diagnostic complement that helps surface interpretable behavioral regularities.

Association rule mining further supports this interpretation by identifying stable co-occurrence patterns between engagement behaviors and task completion. These rules do not imply causality but provide actionable signals that can inform instructional monitoring and early-warning systems.

Advancing the Research Gap

This study responds directly to a persistent gap in learning analytics research: the disconnect between cognitive load theory and scalable LMS-based analytics. Existing frameworks often emphasize predictive accuracy without theoretical interpretability or rely on cognitive theory without feasible measurement on a scale. The proposed nine-step methodology bridges this divide by embedding CLT constructs directly into the analytics pipeline, linking raw LMS events to cognitive load dimensions and learning outcomes in a reproducible manner.

The contribution is therefore methodological rather than merely confirmatory. By demonstrating how Moodle log data can support theory-informed inference when systematically operationalized, the study advances learning analytics from descriptive monitoring toward explanatory and cognitively grounded analysis.

Practical Implications for Design and Policy

The findings suggest that modest, theory-driven interface adjustments, such as structured navigation, guided examples, and collaborative prompts, can yield measurable cognitive and performance benefits without requiring wholesale system redesign. For educators and instructional designers, the study provides concrete mappings between interface features and cognitive load dimensions, supporting more intentional pedagogical alignment. For LMS developers and institutional leaders, the methodology illustrates how existing log data can be repurposed into cognitively meaningful indicators that support scalable monitoring and adaptive intervention, particularly in resource-constrained contexts.

Limitations and Future Research Directions

Several limitations warrant consideration. The study was conducted within a single institution and disciplinary context, which may limit generalizability. Cognitive load was inferred through behavioral proxies rather than measured directly, and the experimental duration was relatively short. Future research should integrate physiological or self-reported measures for triangulation, extend the methodology across institutions and disciplines, and explore real-time adaptive systems that respond dynamically to inferred cognitive load states.

Longitudinal studies may also clarify whether the observed benefits translate into sustained learning gains and schema development over time.

Conclusion

This study proposed and validated a nine-step analytics methodology that systematically integrates cognitive load theory into Moodle-based learning analytics. Unlike existing LMS analytics frameworks that primarily describe engagement patterns, the proposed approach embeds extraneous, intrinsic, and germane cognitive load constructs directly into the analytics pipeline using theory-grounded Moodle log proxies. In doing so, it extends prior frameworks from descriptive monitoring toward explanatory and diagnostic analytics that connect learner behavior, cognitive processing, and performance.

Empirically, the findings demonstrate that cognitive load-informed scaffolds are associated with meaningful differences in learner engagement, behavioral patterns, and performance outcomes in authentic online learning environments. Methodologically, integrating statistical analysis, machine learning, clustering, and association rule mining into a single coherent workflow provides a robust, reproducible foundation for cognitively grounded learning analytics.

Overall, this study advances learning analytics by demonstrating how cognitive load theory can be operationalized at scale using unobstrusive LMS data, offering a transferable framework for the design of cognitively informed and adaptive online learning environments.

REFERENCES

Abeysekera, I., Sunga, E., Gonzales, A., & David, R. (2024). The effect of cognitive load on learning memory of online learning accounting students in the Philippines. *Sustainability, 16*(4), 1686. https://doi.org/10.3390/SU16041686

Ahmed, S. K. (2024). How to choose a sampling technique and determine sample size for research: A simplified guide for researchers. *Oral Oncology Reports*, *12*, 100662. https://doi.org/10.1016/J.OOR.2024.100662

Al-Kindi, I., Al-Khanjari, Z., Jamoussi, Y., Al-Kindi, I. R., Al, S., Al, K., Squ, S., & Muscat, O. (2022). Extracting student patterns from the Moodle log file Moodle course: A case study. *International Journal of Evaluation and Research in Education*, *11*(2), 917–926. https://doi.org/10.11591/ijere.v11i2.23242

Álvarez-Méndez, A. M., Angulo Carrera, M. T., Cristóbal Barrios, J., Bravo-Llatas, M. C., & Álvarez Vázquez, M. P. (2020). Application of data mining in Moodle platform for the analysis of the academic performance of a compulsory subject in university students. *Proceedings 14th International Technology, Education and Development Conference, Valencia, Spain*, 984–992. https://doi.org/10.21125/inted.2020.0355

Bănuţ, M., & Andronache, D. (2023). Students' cognitive load in online education, under the lens of learning theories. *Studia Universitatis Babeş-Bolyai Psychologia-Paedagogia*, *68*(2), 111–130. https://doi.org/10.24193/SUBBPSYPED.2023.2.06

Blayney, P., Kalyuga, S., & Sweller, J. (2015). Using cognitive load theory to tailor instruction to levels of accounting students' expertise. *Journal of Educational Technology & Society*, *18*(4), 199–210. http://www.jstor.org/stable/jeductechsoci.18.4.199

Casteel, A., & Bridier, N. L. (2021). Describing populations and samples in doctoral student research. *International Journal of Doctoral Studies*, *16*, 339–362. https://doi.org/10.28945/4766

Clark, R., & Mayer, R. (2016). e-Learning and the science of instruction. In D. Matthew, K. Michael, T. Rebecca, & l Z. Michae (Eds.), *Sustainability (Switzerland)* (Third, Vol. 11, Issue 1). Pfeiffer. https://doi.org/10.1002/9781119239086

Cohen, L., Manion, L., & Morrison, K. (2007). Experiments, quasi-experiments, single-case research, and meta-analysis. *Research Methods in Education (6th ed.).* London, UK: Routledge. https://doi.org/10.4324/9780203029053-23

Conijn, R., Snijders, C., Kleingeld, A., & Matzat, U. (2017). Predicting student performance from LMS data: A comparison of 17 blended courses using Moodle LMS. *IEEE Transactions on Learning Technologies*, *10*(1), 17–29. https://doi.org/10.1109/TLT.2016.2616312

Cunha, A., & Figueira, Á. (2021). Analysing students' interaction sequences on Moodle to predict academic performance. *Learning Analytics for Smart Learning Environments*, *3024*, 31–37. https://ceur-ws.org/Vol-3024/paper5.pdf

Fenu, G., Marras, M., & Meles, M. (2017). A learning analytics tool for usability assessment in Moodle environments. *Journal of E-Learning and Knowledge Society*, *13*(3), 23–34. https://doi.org/10.20368/1971-8829/1388

Garrison, D. R., Anderson, T., & Archer, W. (1999). Critical inquiry in a text-based environment: Computer conferencing in higher education. *The Internet and Higher Education*, *2*(2–3), 87–105. https://doi.org/10.1016/S1096-7516(00)00016-6

Krishnan, R., Nair, S., Saamuel, B. S., Justin, S., Iwendi, C., Biamba, C., & Ibeke, E. (2022). Smart analysis of learners performance using learning analytics for improving academic progression: A case study model. *Sustainability*, *14*(6), 3378. https://doi.org/10.3390/su14063378

Minkley, N., Xu, K. M., & Krell, M. (2021). Analyzing relationships between causal and assessment factors of cognitive load: Associations between objective and subjective measures of cognitive load, stress, interest, and self-concept. *Frontiers in Education*, *6*, 632907. https://doi.org/10.3389/feduc.2021.632907

Moodle. (n.d.a). *Analytics*. https://docs.moodle.org/500/en/Analytics

Moodle. (n.d.b). *Learning analytics indicators*. https://docs.moodle.org/310/en/Learning_analytics_indicators#Course_level_cognitive_depth_indicators

Orru, G., & Longo, L. (2019). The evolution of cognitive load theory and the measurement of its intrinsic, extraneous, and germane loads: A review. In L. Longo & M. Leva (Eds.), *Human mental workload: Models and applications* (pp. 23–48). Springer. https://doi.org/10.1007/978-3-030-14273-5_3

Rachel, V., Sudhamathy, G., & Parthasarathy, M. (2018). Analytics on Moodle data using R Package for enhanced learning management. *International Journal of Applied Engineering Research*, *13*(22), 15580–15610. https://www.ripublication.com/ijaer18/ijaerv13n22_19.pdf

Renkl, A., & Atkinson, R. K. (2002). Learning from examples: Fostering self-explanations in computer-based learning environments. *Interactive Learning Environments*, *10*(2), 105–119. https://doi.org/10.1076/ilee.10.2.105.7441

Rotelli, D., & Monreale, A. (2021). *Making sense of Moodle log data.* PsyArXiv. http://arxiv.org/abs/2106.11071

Shum, S. B., & Ferguson, R. (2012). Social learning analytics. *Journal of Educational Technology & Society*, *15*(3), 3–26. http://www.jstor.org/stable/jeductechsoci.15.3.3

Siemens, G., & Long, P. D. (2014). Penetrating the fog: Analytics in learning and education. *Italian Journal of Educational Technology*, *22*(3), 132–137. https://doi.org/10.17471/2499-4324/195

Skulmowski, A., & Xu, K. M. (2022). Understanding cognitive load in digital and online learning: A new perspective on extraneous cognitive load. *Educational Psychology Review*, *34*, 171–196. https://doi.org/10.1007/s10648-021-09624-7

Sun, J. C.-Y., Liu, Y., Lin, X., & Hu, X. (2023). Temporal learning analytics to explore traces of self-regulated learning behaviors and their associations with learning performance, cognitive load, and student engagement in an asynchronous online course. *Frontiers in Psychology*, *13*, 1096337. https://doi.org/10.3389/fpsyg.2022.1096337

Sweller, J. (2018). Measuring cognitive load. *Perspectives on Medical Education*, *7*(1), 1–2. https://doi.org/10.1007/s40037-017-0395-4

Sweller, J., Ayres, P., & Kalyuga, S. (2011a). *Cognitive load theory*. Springer. https://doi.org/10.1007/978-1-4419-8126-4

Sweller, J., Ayres, P., & Kalyuga, S. (2011b). The goal-free effect. *Cognitive load theory* (pp. 89–98). Springer. https://doi.org/10.1007/978-1-4419-8126-4_7

Sweller, J., van Merriënboer, J. J. G., & Paas, F. (2019). Cognitive architecture and instructional design: 20 years later. *Educational Psychology Review*, *31*(2), 261–292. https://doi.org/10.1007/S10648-019-09465-5

Tlili, A., Denden, M., Essalmi, F., Jemni, M., Chang, M., Kinshuk, & Chen, N. S. (2023). Automatic modeling learner's personality using a learning analytics approach in an intelligent Moodle learning platform. *Interactive Learning Environments*, *31*(5), 2529–2543. https://doi.org/10.1080/10494820.2019.1636084 van Mierlo, C. M., Jarodzka, H., Kirschner, F., & Kirschner, P. A. (2012). Cognitive load theory in e-learning. In Z. Yan (Ed.), *Encyclopedia of cyber behavior* (pp. 1178-1211). IGI Global Scientific Publishing. https://doi.org/10.4018/978-1-4666-0315-8.ch097

Yen, C.-H., Chen, I.-C., Lai, S.-C., & Chuang, Y.-R. (2015). An analytics-based approach to managing cognitive load by using log data of learning management systems and footprints of social media. *Journal of Educational Technology & Society*, *18*(4), 141–158. http://www.jstor.org/stable/jeductechsoci.18.4.141

APPENDICES

APPENDIX A: MOODLE LOG EXTRACTION AND DATA LOADING PROCEDURES (STEP 1)

This appendix details the procedures used to extract raw data learner interaction data from the Moodle Learning Management System and load them into the analytics environment (Figure A1). Table A1 presents the Moodle log header fields used in this study, while Table A2 summarizes the records contained in the downloaded Excel files. Code excerpt A1 provides a representative Python snippet illustrating how the Excel data were loaded into the Pandas analysis environment.

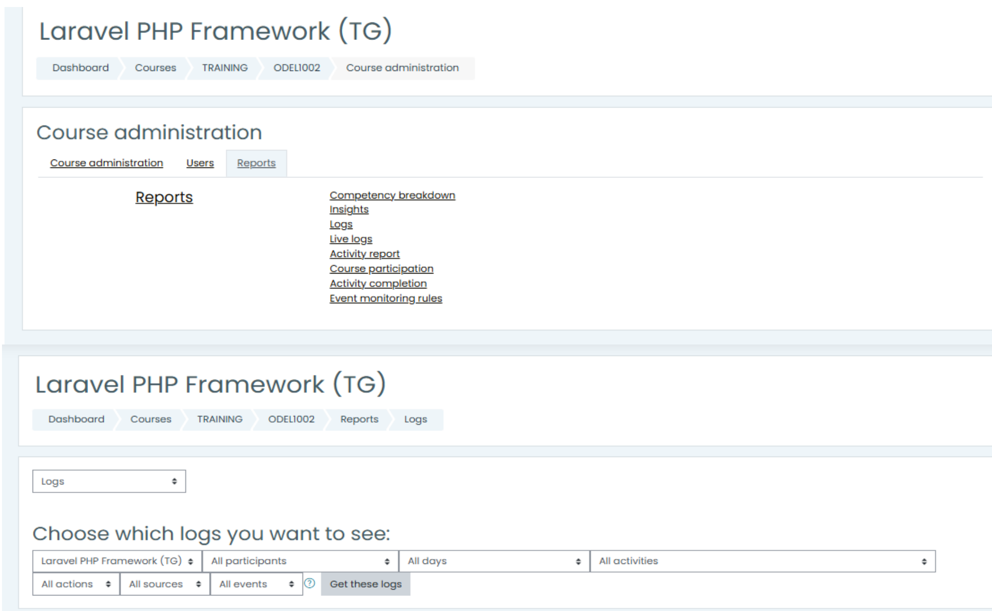

Figure A1. Procedure for downloading Excel file from Moodle LMS

Table A1. Moodle log file header fields

Header name	Description
Time	Timestamp of the log record
User full name	User's full name
Affected user	Affected user's full name
Event context	Context of the event
Component	Component producing the log record.
Event name	Name of the event
Description	Event description
Origin	Origin of the log record (client/ web server)
IP address	The IP address of the device through which the user logged in

Table A2. Summary of the records contained in the downloaded Moodle Excel file

Time	User full name	Affected user	Event context	Component	Event name	Description	Origin	IP address
2/06/23, 01:20	OENGA PETE	-	Course: Advanced Web Programming Group 1	System	Course viewed	The user with id '2928' vie	web	197.232.55.242
1/06/23, 09:40	NGEI LARRY	-	URL: Session 3 Web Conference	URL	Course module view	The user with id '3604' vie	web	197.232.26.117
1/06/23, 09:40	NGEI LARRY	-	Course: Advanced Web Programming Group 1	System	Course viewed	The user with id '3604' vie	web	197.232.26.117
1/06/23, 09:40	NGEI LARRY	-	Course: Advanced Web Programming Group 1	System	Course viewed	The user with id '3604' vie	web	197.232.26.117
1/06/23, 08:53	NGEI LARRY	-	Course: Advanced Web Programming Group 1	System	Course viewed	The user with id '3604' vie	web	197.232.26.117
1/06/23, 08:35	REAGAN ATH	-	URL: LESSON 2 RECORDED WEB CONF	URL	Course module view	The user with id '938' view	web	41.90.217.241
1/06/23, 08:35	REAGAN ATH	-	File: Session 2: Routes, Controllers and Views	File	Course module view	The user with id '938' view	web	41.90.217.241
1/06/23, 08:34	REAGAN ATH	-	Course: Advanced Web Programming Group 1	System	Course viewed	The user with id '938' view	web	41.90.217.241
1/06/23, 08:34	AMIANI PAUI	-	Course: Advanced Web Programming Group 1	System	Course viewed	The user with id '940' view	web	197.181.114.170
1/06/23, 08:34	AMIANI PAUI	-	File: Session 4-Class Notes	File	Course module view	The user with id '940' view	web	197.181.114.170
1/06/23, 08:34	AMIANI PAUI	-	Course: Advanced Web Programming Group 1	System	Course viewed	The user with id '940' view	web	197.181.114.170
1/06/23, 08:34	AMIANI PAUI	-	Course: Advanced Web Programming Group 1	System	Course viewed	The user with id '940' view	web	197.181.114.170
1/06/23, 08:34	REAGAN ATH	-	File: Lesson 1	File	Course module view	The user with id '938' view	web	41.90.217.241
1/06/23, 08:34	AMIANI PAUI	-	File: Session 3	File	Course module view	The user with id '940' view	web	197.181.114.170
1/06/23, 08:34	AMIANI PAUI	-	Course: Advanced Web Programming Group 1	System	Course viewed	The user with id '940' view	web	197.181.114.170
1/06/23, 08:34	REAGAN ATH	-	Course: Advanced Web Programming Group 1	System	Course viewed	The user with id '938' view	web	41.90.217.241
1/06/23, 08:34	AMIANI PAUI	-	Course: Advanced Web Programming Group 1	System	Course viewed	The user with id '940' view	web	197.181.114.170
1/06/23, 08:34	AMIANI PAUI	-	Course: Advanced Web Programming Group 1	System	Course viewed	The user with id '940' view	web	197.181.114.170
1/06/23, 08:34	AMIANI PAUI	-	Course: Advanced Web Programming Group 1	System	Course viewed	The user with id '940' view	web	197.181.114.170
1/06/23, 08:34	REAGAN ATH	-	Course: Advanced Web Programming Group 1	System	Course viewed	The user with id '938' view	web	41.90.217.241
1/06/23, 08:33	CURNELIOUS	-	Course: Advanced Web Programming Group 1	System	Course viewed	The user with id '2293' vie	web	154.159.254.68

Code excerpt A1. Python snippet for loading Excel files into Pandas

```python
import pandas as pd
import numpy as np  # Import NumPy

# Load your treatment group data into a DataFrame
treatment_data = pd.read_csv('Treatment-lms-logs.csv')
```

APPENDIX B: DATA PREPROCESSING AND CLEANING PROCEDURES (STEP 2)

This appendix documents the data preprocessing procedures applied to the raw Moodle logs to ensure data quality and analytical suitability. Table B1 summarizes the learner attributes extracted for subsequent analysis, while Code excerpt B1 shows code for restructuring the datasets using pivot_table(). Code excerpt B2 illustrates the Python routines used for duplicate removal and parsing timestamps. Table B2 reports a sample of a pre-processed dataset.

Table B1. Learner-level attributes derived from Moodle logs for further analysis

Attribute name	Description
User full name	Student name
Logins	Total number of logins (visits) by the student
Course module viewed	Total number of course module views by the student
Course viewed	Total number of course views by the student
Quiz attempt started	Total number of quiz attempts started by the student
Quiz attempt viewed	Total number of quiz attempts viewed by the student
Quiz attempt submitted	Total number of quiz attempts submitted by the student
Time spent on the quiz	Total time taken to complete the quiz by the student
Course activity completion	Total number of course activities completed by the student
Quiz grade/30	Total number of Quiz scores by the student

Code excerpt B1. Python code snippet using pivot_table() for restructuring the dataset.

```
# Group the data by "User full name" and "Event name" to count each event
event_counts = preprocessed_data.groupby(['User full name', 'Event name'])['Time Stamp'].count().reset_index()

# Pivot the event counts to create a wide format table
event_counts_pivot = event_counts.pivot(index='User full name', columns='Event name', values='Time Stamp')
```


```
# Pivot the event counts to create a wide format table
event_counts_pivot = event_counts.pivot(index='User full name', columns='Event name', values='Time Stamp')
```


Code excerpt B2. Code snippet for removing duplicates and parsing timestamps

```
# Step 1: Data Filtering
# Remove duplicates
treatment_data = treatment_data.drop_duplicates()

# Remove logs with no usernames
treatment_data = treatment_data.dropna(subset=['User full name'])

# Remove admin user full names
admin_usernames = ['Tum Administrator', 'Dr. Kennedy Hadullo', 'Musyimi Samuel']
treatment_data = treatment_data[~treatment_data['User full name'].isin(admin_usernames)]

# Remove client-generated and system component logs
#treatment_data = treatment_data[~treatment_data['Component'].isin(['client', 'system'])]

# Step 2: Creating the Time Stamp field
# Assuming your timestamp field is named 'Time'
treatment_data['Time Stamp'] = pd.to_datetime(treatment_data['Time'])
```

Table B2. Showing part of the pre-processed dataset

Student_Id	Course module viewed	Course viewed	Quiz attempt started	Quiz attempt submitted	Quiz attempt viewed	Grand Total	Performance: Average Time Spent in each Activity
std1	12	1	1	1	1	16	0.003109815
std2	1	3	1	1	1	7	0.001360544
std3	13	48	1	1	1	64	0.012439261
std4	2	6	1	1	1	11	0.002137998
std5	8	14	1	1	1	25	0.004859086
std6	14	39	1	1	1	56	0.010884354
std7	7	223	1	1	1	233	0.045286686
std8	1	1	1	1	1	5	0.000971817
std9	32	98	1	1	1	133	0.02585034
std10	2	23	1	1	1	28	0.005442177
std11	13	21	1	1	1	37	0.007191448
std13	17	34	1	1	1	54	0.010495627
std14	1	1	1	1	1	5	0.000971817
std15	1	23	1	1	1	27	0.005247813
std16	5	6	1	1	1	14	0.002721088
std17	12	1	1	1	1	16	0.003109815
std18	25	34	1	1	11	72	0.013994169
std19	12	9	1	1	1	24	0.004664723
std20	4	5	1	1	1	12	0.002332362

Appendix C: Data Transformation and Feature Engineering (Step 3)

This appendix describes the data transformation procedures used to derive analytically meaningful variables from Moodle log fields. Code Excerpt C1 demonstrates the feature engineering operations (imputation and normalization procedures) implemented to prepare variables for cognitive load mapping and statistical analysis.

Code excerpt C1. Missing value imputation and normalization

```
import pandas as pd

# Load your preprocessed data into a DataFrame
preprocessed_data = pd.read_csv('preprocessed_Treatment-lms-logs.csv')

# Assuming "Time Stamp" is already in datetime format, if not, convert it first
# preprocessed_data['Time Stamp'] = pd.to_datetime(preprocessed_data['Time Stamp'])

# Group the data by "User full name" and count the number of logins
login_counts = preprocessed_data.groupby('User full name')['Time Stamp'].count().reset_index()

# Rename the "Time Stamp" column to "Logins"
login_counts = login_counts.rename(columns={'Time Stamp': 'Logins'})

# Group the data by "User full name" and "Event name" to count each event
event_counts = preprocessed_data.groupby(['User full name', 'Event name'])['Time Stamp'].count().reset_index()

# Pivot the event counts to create a wide format table
event_counts_pivot = event_counts.pivot(index='User full name', columns='Event name', values='Time Stamp')

# Fill NaN values with the mean for each column
event_counts_pivot = event_counts_pivot.apply(lambda col: col.fillna(col.mean()), axis=0)
```

Appendix D: Mapping Moodle Log Events to Cognitive Load and Performance Constructs (Step 4)

This appendix provides the detailed mapping framework used to associate Moodle log indicators with extraneous, intrinsic, and germane cognitive load dimensions, as well as learner performance. Table D1 presents the full mapping matrix linking log events to cognitive load constructs, while Table D2 summarizes the theoretical justification for each mapping based on cognitive load theory and prior literature. Table D3 presents the operational thresholds for classifying interaction frequency levels. Code Excerpt D1 presents the function for mapping Moodle learner attributes to cognitive load dimensions.

Table D1. Mapping moodle learner attributes to cognitive load and performance constructs

Moodle learner attribute	Cognitive load/ performance construct
Course module viewed	Extraneous Load
Course viewed	
Quiz attempt started	Intrinsic Load
Quiz attempt viewed	
Quiz attempt submitted	Germane Load
Time spent on the quiz	
Course activity completion	Performance
Quiz grades	

Table D2. Theoretical justification for each mapping based on cognitive load theory and prior literature

Moodle log indicator	Cognitive load dimension	Theoretical justification	Key supporting literature
Course viewed	Extraneous Cognitive load	Repeated course-level navigation reflects interface-related effort rather than learning activity and may indicate disorientation or inefficient structure that increases extraneous processing.	Skulmowski and Xu (2022); Sun et al.(2023); Blayney et al. (2015)

Moodle log indicator	Cognitive load dimension	Theoretical justification	Key supporting literature
Course module viewed		Frequent module navigation suggests fragmented content access and interface demands that impose cognitive effort unrelated to schema construction.	Skulmowski and Xu (2022); Sun et al.(2023)
Quiz attempt started	Intrinsic cognitive load	Initiating quizzes reflects engagement with the task complexity inherent in the learning material and learners' attempt to process intrinsic demands.	Sweller (2018); Orru and Longo (2019b)
Quiz attempt viewed	Intrinsic cognitive load	Viewing quiz attempts without submission indicates cognitive appraisal of task requirements and problem structure linked to intrinsic load.	Abeysekera et al. (2024); Yen et al. (2015)
Quiz attempt submitted	Germane cognitive load	Submitting quizzes reflects active application of knowledge and schema construction, characteristic of germane cognitive processing.	Sweller et al. (2011b); Skulmowski and Xu (2022)
Course activity completion	Learning outcome (associated with germane load)	Completing learning activities signals sustained cognitive engagement and successful integration of instructional content.	Garrison et al. (1999); Álvarez-Méndez et al.(2020); Al-Kindi et al. (2022)
Grade/quiz score	Learning outcome (associated with germane load)	Performance outcomes indirectly validate effective germane processing resulting from meaningful cognitive investment.	Sun et al.(2023); Krishnan et al. (2022); Al-Kindi et al. (2022)

Table D3. Operational thresholds for classifying interaction frequency levels

Moodle log indicator	Low interaction threshold	High interaction threshold	Threshold deviation method
Course viewed	≤ 1 view	≥ 3 views	Lower and upper quartiles of observed d
Course module viewed	≤ 2 views	≥ 5 views	Interquartile range-based cut-off
Quiz attempt started	0 attempts	≥ 1 attempt	Binary task engagement criterion
Quiz attempt viewed	≤ 1 view	≥ 2 views	Median split of interaction frequency
Quiz attempt submitted	0 submissions	≥ 1 submission	Task completion indicator
Course activity competition	≤ 50% activities completed	≥ 80% activities	Percent completion thresholds
Grade/quiz score	≤ 50%	≥ 70%	Institutional grading benchmark

Code excerpt D1. Mapping of moodle learner attributes to cognitive load

```
#Cognitive Load to Lms indicator mapping
import pandas as pd
from IPython.display import display

# Function to map Moodle LMS indicators to Cognitive Loads
def map_labels(indicator):
    extraneous_load = ['Course module viewed', 'Course viewed']
    intrinsic_load = ['Quiz attempt started', 'Quiz attempt viewed']
    germane_load = ['Quiz attempt submitted', 'Time Spent on Quiz Activity']
    performance = ['Course activity completion', 'Grade/30']

    if indicator in extraneous_load:
        return 'Extraneous Load'
    elif indicator in intrinsic_load:
        return 'Intrinsic Load'
    elif indicator in germane_load:
        return 'Germane Load'
    elif indicator in performance:
        return 'Performance'
    else:
        return 'Other'

# Read and format data from Excel file
def read_and_format_data(file_name):
    try:
        df = pd.read_excel(file_name)
        if df.empty:
            print("Warning: DataFrame is empty.")
            return None
        indicator_counts = df.drop(columns=['Student']).sum().reset_index()
        indicator_counts.columns = ['Moodle LMS indicator', 'Count']
        indicator_counts['Label'] = indicator_counts['Moodle LMS indicator'].apply(map_labels)
        return indicator_counts
    except Exception as e:
        print(f"Error occurred while processing file: {str(e)}")
        return None

# Read and format data for treatment group
treatment_file_name = 'wk4-tg-indicators-rules.xlsx'
treatment_data = read_and_format_data(treatment_file_name)

# Print formatted data for treatment group
if treatment_data is not None:
    print("Treatment Group:")
    display(treatment_data)
```

Appendix E: Exploratory Data Analysis Outputs (Steps 5)

This appendix presents the supplementary exploratory analysis outputs used to examine distributions, trends, and relationships among cognitive load indicators and performance measures. Figure E1 details the correctional matrices among the variables, Figure E2 reports group-level comparisons used to inform subsequent inferential analyses, while Tables E1 and E2 present group-level summary descriptive statistics.

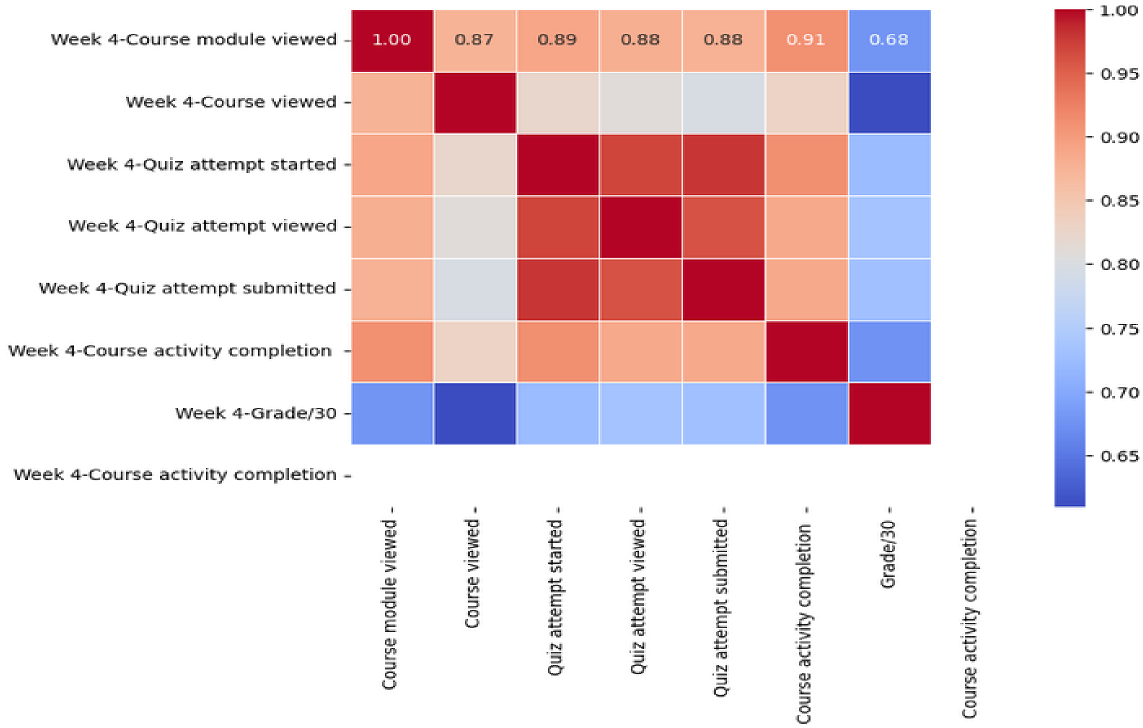

Figure E1. Correlation matrices with coefficients among all variables

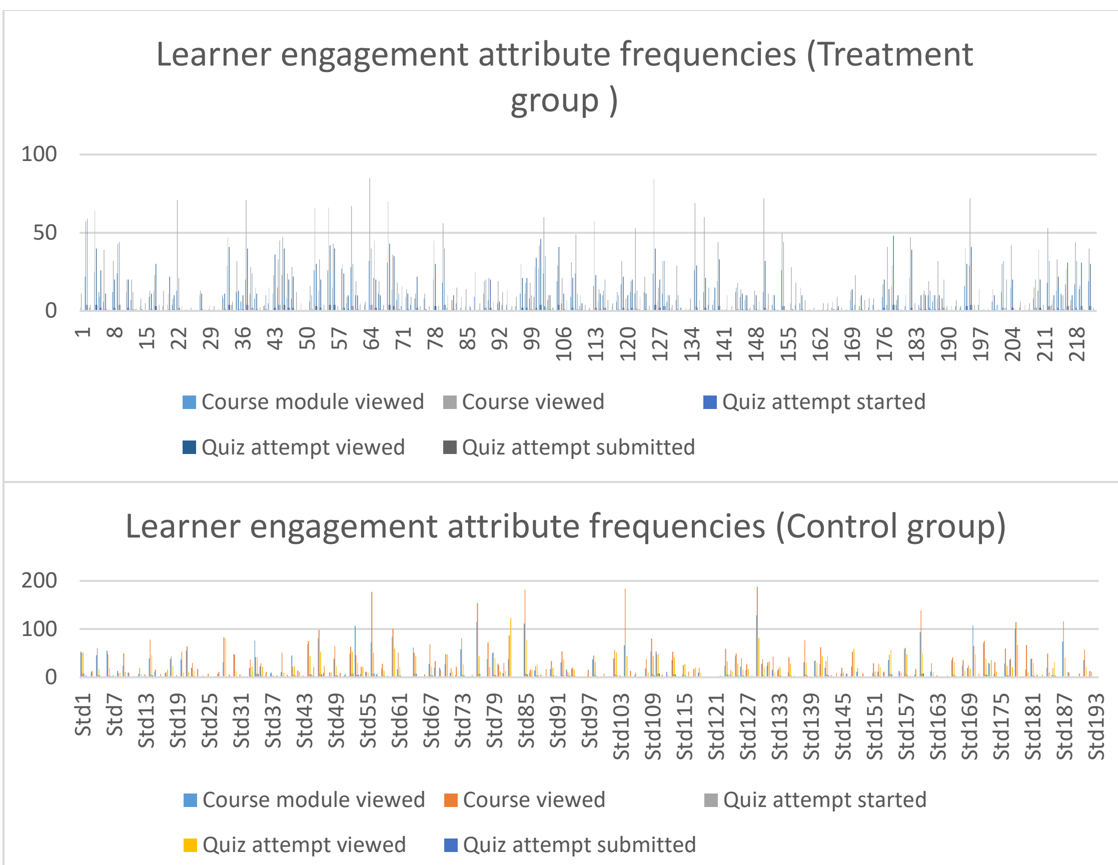

Figure E2. Group-level learner engagement comparisons

Table E1. Descriptive statistics for the control group

	Course module viewed	Course viewed	Quiz attempt started	Quiz attempt viewed	Quiz attempt submitted	Course activity completion	Grade/30
count	183.000000	183.000000	183.000000	183.000000	183.000000	183.000000	183.000000
mean	4.371585	10.836066	0.377049	4.322404	0.355191	4.218579	7.666667
std	2.958256	11.538536	0.485977	5.240364	0.479884	1.571128	10.894264
min	1.000000	1.000000	0.000000	0.000000	0.000000	2.000000	0.000000
25%	3.000000	3.000000	0.000000	0.000000	0.000000	3.000000	0.000000
50%	3.000000	7.000000	0.000000	1.000000	0.000000	4.000000	0.000000
75%	6.000000	14.000000	1.000000	10.000000	1.000000	6.000000	21.000000
max	17.000000	62.000000	1.000000	22.000000	1.000000	8.000000	27.000000

Table E2. Descriptive statistics for the treatment group

	Course module viewed	Course viewed	Quiz attempt started	Quiz attempt viewed	Quiz attempt submitted	Course activity completion	Grade/30
count	162.000000	162.000000	162.000000	162.000000	162.000000	162.000000	162.000000
mean	8.086420	12.993827	0.364198	4.950617	0.358025	5.685185	7.537037
std	9.464149	14.303063	0.482697	6.372303	0.480906	4.815183	10.530723
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
25%	1.000000	3.000000	0.000000	0.000000	0.000000	1.000000	0.000000
50%	4.000000	8.000000	0.000000	0.000000	0.000000	4.500000	0.000000
75%	12.000000	18.750000	1.000000	10.000000	1.000000	10.000000	21.000000
max	43.000000	114.000000	1.000000	43.000000	1.000000	19.000000	27.000000

Appendix F: Statistical Analysis Procedures and Supplementary Results (Step 6)

This appendix details the statistical analysis procedures applied to test relationships between cognitive load dimensions and learner performance. Code Excerpt F1 provides representative Python code used to implement the non-parametric and regression-based analyses.

Code excerpt F1. Python code used to implement the non-parametric and regression analyses

```
#Inference Statistics
#Step 1: Perform Non-Parametric Tests
import pandas as pd
import numpy as np
from scipy.stats import mannwhitneyu
import seaborn as sns
import matplotlib.pyplot as plt

# Assuming df_combined is your dataframe with 'group', 'week', and cognitive load/performance indicators

variables = ['Course module viewed', 'Course viewed', 'Quiz attempt started',
             'Quiz attempt viewed', 'Quiz attempt submitted',
             'Course activity completion', 'Grade/30']

# Function to perform Mann-Whitney U Test
def mann_whitney_test(var):
    treatment_data = df_combined[df_combined['group'] == 'Treatment'][var]
    control_data = df_combined[df_combined['group'] == 'Control'][var]
    stat, p = mannwhitneyu(treatment_data, control_data, alternative='two-sided')
    return stat, p

# Perform tests and collect results
test_results = []
for var in variables:
    stat, p = mann_whitney_test(var)
    test_results.append({'Variable': var, 'U-Statistic': stat, 'p-value': p})

# Convert results to DataFrame
test_results_df = pd.DataFrame(test_results)
print(test_results_df)
```

Appendix G: Clustering Analysis Procedures (Step 7)

This appendix documents the clustering analysis procedures used to identify learner behavior profiles based on cognitive load indicators. Code Excerpt G1 illustrates the clustering workflow implemented using Python-based machine learning libraries, while Figure G1 presents the silhouette score distributions.

Code Excerpt G1. K-means clustering and silhouette evaluation

```
#clustring process
import pandas as pd
from sklearn.preprocessing import StandardScaler
from sklearn.cluster import KMeans
from sklearn.metrics import silhouette_score
import matplotlib.pyplot as plt

# Dictionary containing file names for each week for the treatment group
treatment_file_names = {
    'Week 1': 'wk1-tg-indicators.xlsx',
    'Week 2': 'wk2-tg-indicators.xlsx',
    'Week 3': 'wk3-tg-indicators.xlsx',
    'Week 4': 'wk4-tg-indicators.xlsx'
}

# Dictionary containing file names for each week for the control group
control_file_names = {
    'Week 1': 'wk1-cg-indicators.xlsx',
    'Week 2': 'wk2-cg-indicators.xlsx',
    'Week 3': 'wk3-cg-indicators.xlsx',
    'Week 4': 'wk4-cg-indicators.xlsx'
}

# Function to calculate silhouette scores for a given dataset and range of clusters
def calculate_silhouette_scores(data, cluster_range):
    silhouette_scores = []
    for n_clusters in cluster_range:
        kmeans = KMeans(n_clusters=n_clusters, random_state=42)
        cluster_labels = kmeans.fit_predict(data)
        silhouette_avg = silhouette_score(data, cluster_labels)
        silhouette_scores.append(silhouette_avg)
    return silhouette_scores

# Perform K-means clustering for different numbers of clusters
treatment_silhouette_scores_week = []
for n_clusters in range(2, 11):
    kmeans = KMeans(n_clusters=n_clusters, random_state=42)
    cluster_labels = kmeans.fit_predict(treatment_scaled)
    silhouette_avg = silhouette_score(treatment_scaled, cluster_labels)
    treatment_silhouette_scores_week.append(silhouette_avg)

treatment_silhouette_scores.append((week, treatment_silhouette_scores_week))
```

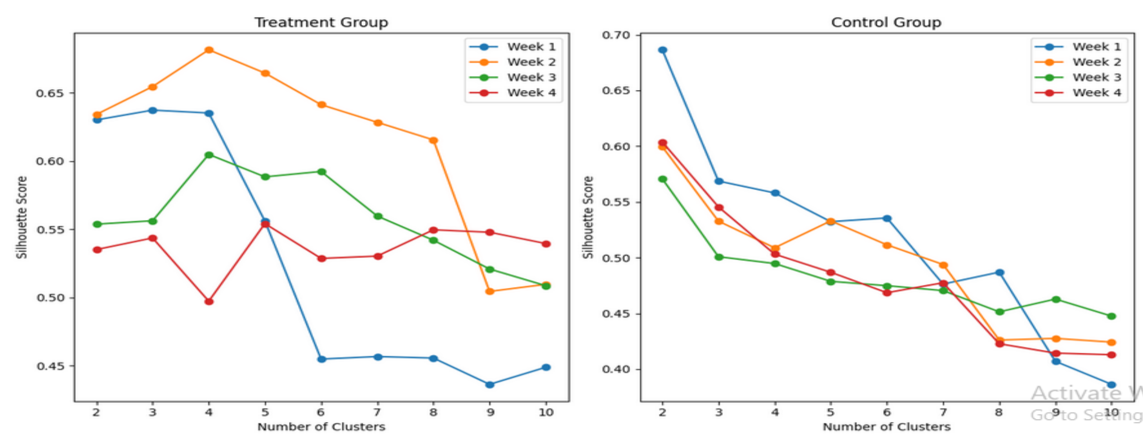

Figure G1. Silhouette scores for k = 2…10.

APPENDIX H: CLASSIFICATION MODEL CONFIGURATION AND PERFORMANCE METRICS (STEP 8)

This appendix presents supplementary details on classification models used to predict learner performance from cognitive load indicators. Tables H1 and H2 provide sample categorization, while Figures H1, H2, and H3 report model performance metrics. Code Excerpt H1 demonstrates the model training and evaluation procedures referenced in the results section.

Table H1. Part of the treatment group categorized learners

Student	Time spent on quiz activity (minutes)	Learner category
std1	2.833333333	Medium
std2	0	Absent/Not Attempted
std3	0	Absent/Not Attempted
std4	5.416666667	High
std5	6.4	High
std6	0	Absent/Not Attempted
std7	0	Absent/Not Attempted
std8	6.8	High
std9	0	Absent/Not Attempted
std10	0	Absent/Not Attempted

Table H2. Part of the control group categorized learners

Student	Time spent on quiz activity (minutes)	Learner category
Std1	0	Absent/Not Attempted
Std2	15.8	High
Std3	0	Absent/Not Attempted
Std4	8.7	High
Std5	0	Absent/Not Attempted
Std6	0	Absent/Not Attempted
Std7	8.716666667	High
Std8	13.01666667	High
Std9	1.466666667	Low
Std10	8.8	High

```
Decision Tree Classifier:
Accuracy: 1.0
Classification Report:
              precision    recall  f1-score   support

           0       1.00      1.00      1.00         9
           1       1.00      1.00      1.00         1
           2       1.00      1.00      1.00         1
           3       1.00      1.00      1.00        22

    accuracy                           1.00        33
   macro avg       1.00      1.00      1.00        33
weighted avg       1.00      1.00      1.00        33

Confusion Matrix:
 [[ 9  0  0  0]
 [ 0  1  0  0]
 [ 0  0  1  0]
 [ 0  0  0 22]]
```

Figure H1. Decision tree classifier model performance

```
Random Forest Classifier:
Accuracy: 1.0
Classification Report:
              precision    recall  f1-score   support

           0       1.00      1.00      1.00         9
           1       1.00      1.00      1.00         1
           2       1.00      1.00      1.00         1
           3       1.00      1.00      1.00        22

    accuracy                           1.00        33
   macro avg       1.00      1.00      1.00        33
weighted avg       1.00      1.00      1.00        33

Confusion Matrix:
 [[ 9  0  0  0]
 [ 0  1  0  0]
 [ 0  0  1  0]
 [ 0  0  0 22]]
```

Figure H2. Random forest classifier model performance

```
Logistic Regression Classifier:
Accuracy: 0.9696969696969697
Classification Report:
              precision    recall  f1-score   support

           0       0.90      1.00      0.95         9
           1       1.00      1.00      1.00         1
           2       0.00      0.00      0.00         1
           3       1.00      1.00      1.00        22

    accuracy                           0.97        33
   macro avg       0.72      0.75      0.74        33
weighted avg       0.94      0.97      0.96        33

Confusion Matrix:
 [[ 9  0  0  0]
 [ 0  1  0  0]
 [ 1  0  0  0]
 [ 0  0  0 22]]
```

Figure H3. Logistic regression classifier model performance

Code Excerpt H1. Classification process

```
#classification models
import pandas as pd
import re
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, classification_report, confusion_matrix
```

```
# Function to process data and perform learner classification
def process_data(file_names, group_name):
    for week, file_name in file_names.items():
        print(f"\n{group_name} - {week}:")
        try:
            data = pd.read_excel(file_name)
            # Rename the column 'student-name' to 'Student'
            data.rename(columns={'student-name': 'Student'}, inplace=True)
            if 'Grade/30' in data.columns:
                # Create 'Performance' column based on 'Grade/30'
                data['Performance'] = pd.cut(data['Grade/30'], bins=[0, 15, 20, 30], labels=['FAIL', 'GOOD', 'EXCELLENT'])
                # Encode 'Performance' column
                label_encoder = LabelEncoder()
                data['Performance'] = label_encoder.fit_transform(data['Performance'])

                # Select features and target variable
                X = data.drop(columns=['Student', 'Performance'])
                y = data['Performance']

                # Check for non-numeric values in features
                non_numeric_cols = X.select_dtypes(exclude=['float64', 'int64']).columns
                if non_numeric_cols.size > 0:
                    X.drop(columns=non_numeric_cols, inplace=True)

                # Split data into train and test sets
                X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

```
# Train Decision Tree classifier
dt_classifier = DecisionTreeClassifier()
dt_classifier.fit(X_train, y_train)
dt_predictions = dt_classifier.predict(X_test)
dt_accuracy = accuracy_score(y_test, dt_predictions)
dt_report = classification_report(y_test, dt_predictions)
dt_conf_matrix = confusion_matrix(y_test, dt_predictions)

print("Decision Tree Classifier:")
print("Accuracy:", dt_accuracy)
print("Classification Report:\n", dt_report)
print("Confusion Matrix:\n", dt_conf_matrix)

# Train Random Forest classifier
rf_classifier = RandomForestClassifier()
rf_classifier.fit(X_train, y_train)
rf_predictions = rf_classifier.predict(X_test)
rf_accuracy = accuracy_score(y_test, rf_predictions)
rf_report = classification_report(y_test, rf_predictions)
rf_conf_matrix = confusion_matrix(y_test, rf_predictions)

print("\nRandom Forest Classifier:")
print("Accuracy:", rf_accuracy)
print("Classification Report:\n", rf_report)
print("Confusion Matrix:\n", rf_conf_matrix)
```

Model performance full metrics

```
Treatment Group - Week 1:
X shape after preprocessing: (162, 7)
X_train shape: (96, 7), y_train shape: (96,)
X_val shape: (33, 7), y_val shape: (33,)
proba_predictions shape for Random Forest: (33, 4)
proba_predictions shape for Support Vector Machine: (33, 4)
proba_predictions shape for Naive Bayes: (33, 4)
proba_predictions shape for K-Nearest Neighbors: (33, 4)
                  Classifier  Training Time (s)  Accuracy  Precision
Recall  \
0            Random Forest            0.158737  0.939394   0.903030
0.939394
1   Support Vector Machine            0.005100  0.939394   0.888889
0.939394
```

```
2            Naive Bayes           0.001998  0.939394   0.888889
0.939394
3     K-Nearest Neighbors          0.001999  0.848485   0.807359
0.848485

   F1-Score       AUC  Error Rate  Support
0  0.916667  0.991935    0.060606     33.0
1  0.911846  0.959677    0.060606     33.0
2  0.911846  0.875000    0.060606     33.0
3  0.818759  0.840667    0.151515     33.0

Treatment Group - Week 2:
X shape after preprocessing: (180, 7)
X_train shape: (108, 7), y_train shape: (108,)
X_val shape: (36, 7), y_val shape: (36,)
proba_predictions shape for Random Forest: (36, 3)
Error processing Week 2 data: Number of classes in y_true not equal to
the number of columns in ‘y_score.’

Treatment Group - Week 3:
X shape after preprocessing: (191, 7)
X_train shape: (114, 7), y_train shape: (114,)
X_val shape: (38, 7), y_val shape: (38,)
proba_predictions shape for Random Forest: (38, 4)
Error processing Week 3 data: Number of classes in y_true not equal to
the number of columns in ‘y_score.’

Treatment Group - Week 4:
X shape after preprocessing: (193, 7)
X_train shape: (115, 7), y_train shape: (115,)
X_val shape: (39, 7), y_val shape: (39,)
proba_predictions shape for Random Forest: (39, 4)
proba_predictions shape for Support Vector Machine: (39, 4)
proba_predictions shape for Naive Bayes: (39, 4)
proba_predictions shape for K-Nearest Neighbors: (39, 4)
               Classifier  Training Time (s)  Accuracy  Precision
Recall  \
0           Random Forest          0.150589  1.000000   1.000000
1.000000
1  Support Vector Machine          0.006149  0.846154   0.873626
0.846154
2            Naive Bayes           0.001999  0.923077   0.940171
0.923077
3     K-Nearest Neighbors          0.003999  0.743590   0.800366
0.743590

   F1-Score       AUC  Error Rate  Support
0  1.000000  1.000000    0.000000     39.0
1  0.828853  0.900769    0.153846     39.0
2  0.924542  0.978439    0.076923     39.0
3  0.762393  0.861066    0.256410     39.0

Control Group - Week 1:
X shape after preprocessing: (183, 7)
X_train shape: (109, 7), y_train shape: (109,)
X_val shape: (37, 7), y_val shape: (37,)
proba_predictions shape for Random Forest: (37, 4)
```

```
proba_predictions shape for Support Vector Machine: (37, 4)
proba_predictions shape for Naive Bayes: (37, 4)
proba_predictions shape for K-Nearest Neighbors: (37, 4)
               Classifier  Training Time (s)  Accuracy  Precision
Recall  \
0           Random Forest           0.151800  0.945946   0.927928
0.945946
1  Support Vector Machine           0.004001  0.918919   0.873874
0.918919
2             Naive Bayes           0.001999  0.810811   0.881596
0.810811
3     K-Nearest Neighbors           0.001999  0.864865   0.888889
0.864865

   F1-Score       AUC  Error Rate  Support
0  0.932432  0.978968    0.054054     37.0
1  0.894349  0.968651    0.081081     37.0
2  0.829580  0.910847    0.189189     37.0
3  0.876245  0.954993    0.135135     37.0

Control Group - Week 2:
X shape after preprocessing: (232, 7)
X_train shape: (138, 7), y_train shape: (138,)
X_val shape: (47, 7), y_val shape: (47,)
proba_predictions shape for Random Forest: (47, 4)
proba_predictions shape for Support Vector Machine: (47, 4)
proba_predictions shape for Naive Bayes: (47, 4)
proba_predictions shape for K-Nearest Neighbors: (47, 4)
               Classifier  Training Time (s)  Accuracy  Precision
Recall  \
0           Random Forest           0.153562  0.978723   0.959574
0.978723
1  Support Vector Machine           0.004997  0.978723   0.959574
0.978723
2             Naive Bayes           0.001996  0.978723   0.959574
0.978723
3     K-Nearest Neighbors           0.002069  0.957447   0.939480
0.957447

   F1-Score       AUC  Error Rate  Support
0  0.968645  0.977323    0.021277     47.0
1  0.968645  0.999269    0.021277     47.0
2  0.968645  0.871711    0.021277     47.0
3  0.947400  0.867786    0.042553     47.0

Control Group - Week 3:
X shape after preprocessing: (214, 7)
X_train shape: (128, 7), y_train shape: (128,)
X_val shape: (43, 7), y_val shape: (43,)
proba_predictions shape for Random Forest: (43, 4)
proba_predictions shape for Support Vector Machine: (43, 4)
proba_predictions shape for Naive Bayes: (43, 4)
proba_predictions shape for K-Nearest Neighbors: (43, 4)
               Classifier  Training Time (s)  Accuracy  Precision
Recall  \
0           Random Forest           0.154660  0.976744   0.955277
0.976744
```

```
1  Support Vector Machine           0.005090  0.976744   0.955603
0.976744
2            Naive Bayes            0.001996  0.976744   0.955277
0.976744
3     K-Nearest Neighbors           0.001999  0.976744   0.955603
0.976744

   F1-Score       AUC  Error Rate  Support
0  0.965581  0.990399    0.023256     43.0
1  0.965670  0.904762    0.023256     43.0
2  0.965581  0.875000    0.023256     43.0
3  0.965670  0.867911    0.023256     43.0

Control Group - Week 4:
X shape after preprocessing: (222, 7)
X_train shape: (132, 7), y_train shape: (132,)
X_val shape: (45, 7), y_val shape: (45,)
proba_predictions shape for Random Forest: (45, 4)
proba_predictions shape for Support Vector Machine: (45, 4)
proba_predictions shape for Naive Bayes: (45, 4)
proba_predictions shape for K-Nearest Neighbors: (45, 4)
               Classifier  Training Time (s)  Accuracy  Precision
Recall  \
0          Random Forest           0.152646  0.933333   0.983333
0.933333
1  Support Vector Machine           0.005996  0.777778   0.780952
0.777778
2            Naive Bayes            0.001982  0.955556   0.966667
0.955556
3     K-Nearest Neighbors           0.002108  0.800000   0.819259
0.800000

   F1-Score       AUC  Error Rate  Support
0  0.933333  1.000000    0.066667     45.0
1  0.770000  0.944893    0.222222     45.0
2  0.958057  1.000000    0.044444     45.0
3  0.798942 0.881739    0.200000     45.0
```

Appendix G: Association Rule Mining Procedures (Step 9)

This appendix details the association rule mining procedures used to identify recurring relationships between cognitive load patterns and learner performance. Code Excerpt I1 illustrates the implementation of the Apriori-based rule mining algorithm.

Code Excerpt I1. Association rules evaluation

```
import pandas as pd
from mlxtend.frequent_patterns import apriori, association_rules

# Step 1: Load the dataset
df = pd.read_excel('wk4-tg-indicators-rules.xlsx')

# Step 2: Display the first few rows of the dataframe
print("First few rows of the dataset:")
print(df.head())

# Step 3: Preprocess the data (if necessary)
# No preprocessing required in this case

# Step 4: Perform Association Rule Mining
# Convert the dataset into a transaction format
transactions = df.applymap(str)

# Perform one-hot encoding
onehot = pd.get_dummies(transactions)

# Apply the Apriori algorithm to find frequent itemsets
frequent_itemsets = apriori(onehot, min_support=0.6, use_colnames=True)

# Generate association rules
rules = association_rules(frequent_itemsets, metric='confidence', min_threshold=0.9)

# Step 5: Display the generated rules
print("\nGenerated Association Rules:")
print(rules)
```

Authors

Thaddeus Ogwoka is a Tutorial Fellow in the Institute of Computing and Informatics at the Technical University of Mombasa, Kenya. He holds a Master's degree in Computer Systems and a Bachelor of Science degree in Information Technology from Jomo Kenyatta University of Agriculture and Technology, Kenya. He is currently a Doctor of Philosophy (PhD) candidate in Information Systems at the University of Nairobi. His research interests lie in ICT for Education research and practice, particularly the use of adaptive interfaces with learner modeling, machine learning, and adaptive web-based learning interfaces that scaffold learners.

Professor Robert Oboko is a Professor of Computer Science in the Department of Computing and Informatics at the University of Nairobi, Kenya. He holds a PhD in Computer Science and an MA in Development Studies from the University of Nairobi, an MSc in Computer Science from the Free University of Brussels, and a Bachelor of Education (Mathematics) from Kenyatta University, Kenya. His research interests focus on ICT for Education research and practice, particularly the use of adaptive interfaces with learner modeling, machine learning, and adaptive web-based learning interfaces to scaffold learners. He also supervises and collaborates with students at all levels on applying artificial intelligence beyond education, including agriculture and health, and is actively involved in research and consultancy in ICT for Development.

Professor Christopher Chepken is an Associate Professor of Computer Science in the Department of Computing and Informatics at the University of Nairobi, Kenya. He holds a PhD in Computer Science from the University of Cape Town, an MSc in Applied Computer Science, and a BSc in Computer Science from the University of Nairobi. His research and consultancy work focus on ICT for Development, with applications in urban commuting challenges, telecommuting, education, health, agriculture, and business process outsourcing, through which he has gained extensive experience in implementing ICT-and development-related initiatives.